```
            bjp=bjp*BIGNI
            bessj=bessj*BIGNI
            sum=sum*BIGNI
        endif
        if(jsum.ne.0)sum=sum+bj     Accumulate the sum.
        jsum=1-jsum                 Change 0 to 1 or vice versa.
        if(j.eq.n)bessj=bjp         Save the unnormalized answer.
    enddo 12
    sum=2.*sum-bj                   Compute (5.5.16)
    bessj=bessj/sum                 and use it to normalize the answer.
endif
if(x.lt.0..and.mod(n,2).eq.1)bessj=-bessj
return
END
```

CITED REFERENCES AND FURTHER READING:

Abramowitz, M., and Stegun, I.A. 1964, *Handbook of Mathematical Functions*, Applied Mathematics Series, Volume 55 (Washington: National Bureau of Standards; reprinted 1968 by Dover Publications, New York), Chapter 9.

Hart, J.F., et al. 1968, *Computer Approximations* (New York: Wiley), §6.8, p. 141. [1]

6.6 Modified Bessel Functions of Integer Order

The modified Bessel functions $I_n(x)$ and $K_n(x)$ are equivalent to the usual Bessel functions J_n and Y_n evaluated for purely imaginary arguments. In detail, the relationship is

$$\begin{aligned} I_n(x) &= (-i)^n J_n(ix) \\ K_n(x) &= \frac{\pi}{2} i^{n+1} [J_n(ix) + iY_n(ix)] \end{aligned} \tag{6.6.1}$$

The particular choice of prefactor and of the linear combination of J_n and Y_n to form K_n are simply choices that make the functions real-valued for real arguments x.

For small arguments $x \ll n$, both $I_n(x)$ and $K_n(x)$ become, asymptotically, simple powers of their argument

$$\begin{aligned} I_n(x) &\approx \frac{1}{n!}\left(\frac{x}{2}\right)^n \qquad n \geq 0 \\ K_0(x) &\approx -\ln(x) \\ K_n(x) &\approx \frac{(n-1)!}{2}\left(\frac{x}{2}\right)^{-n} \qquad n > 0 \end{aligned} \tag{6.6.2}$$

These expressions are virtually identical to those for $J_n(x)$ and $Y_n(x)$ in this region, except for the factor of $-2/\pi$ difference between $Y_n(x)$ and $K_n(x)$. In the region

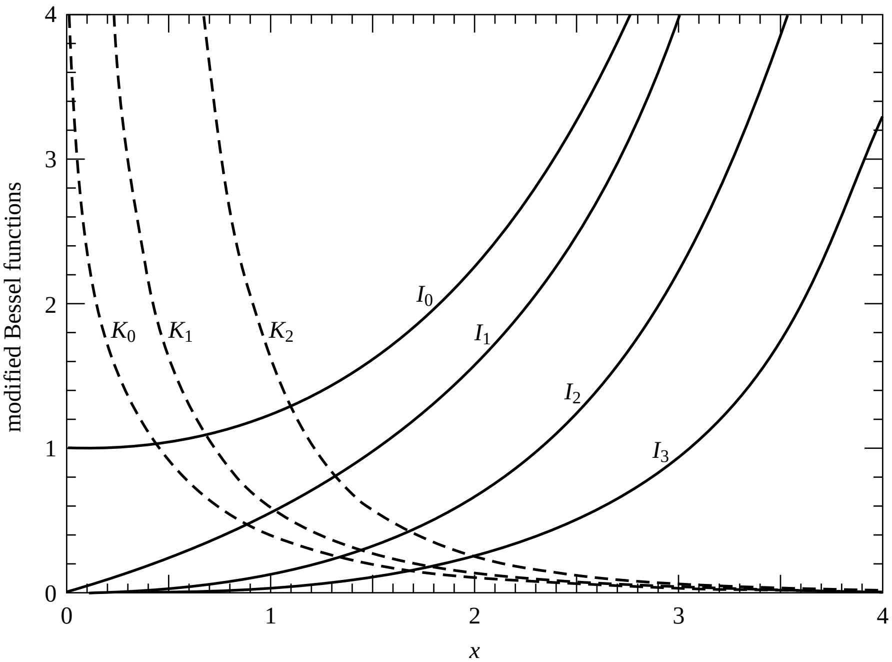

Figure 6.6.1. Modified Bessel functions $I_0(x)$ through $I_3(x)$, $K_0(x)$ through $K_2(x)$.

$x \gg n$, however, the modified functions have quite different behavior than the Bessel functions,

$$I_n(x) \approx \frac{1}{\sqrt{2\pi x}} \exp(x)$$
$$K_n(x) \approx \frac{\pi}{\sqrt{2\pi x}} \exp(-x) \tag{6.6.3}$$

The modified functions evidently have exponential rather than sinusoidal behavior for large arguments (see Figure 6.6.1). The smoothness of the modified Bessel functions, once the exponential factor is removed, makes a simple polynomial approximation of a few terms quite suitable for the functions I_0, I_1, K_0, and K_1. The following routines, based on polynomial coefficients given by Abramowitz and Stegun [1], evaluate these four functions, and will provide the basis for upward recursion for $n > 1$ when $x > n$.

```
      FUNCTION bessi0(x)
      REAL bessi0,x
         Returns the modified Bessel function I0(x) for any real x.
      REAL ax
      DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,
     *     q8,q9,y                        Accumulate polynomials in double precision.
      SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
      DATA p1,p2,p3,p4,p5,p6,p7/1.0d0,3.5156229d0,3.0899424d0,1.2067492d0,
     *     0.2659732d0,0.360768d-1,0.45813d-2/
      DATA q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,0.1328592d-1,
     *     0.225319d-2,-0.157565d-2,0.916281d-2,-0.2057706d-1,
     *     0.2635537d-1,-0.1647633d-1,0.392377d-2/
```

```
      if (abs(x).lt.3.75) then
          y=(x/3.75)**2
          bessi0=p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7)))))
      else
          ax=abs(x)
          y=3.75/ax
          bessi0=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4
*             +y*(q5+y*(q6+y*(q7+y*(q8+y*q9))))))))
      endif
      return
      END
```

```
      FUNCTION bessk0(x)
      REAL bessk0,x
C     USES bessi0
          Returns the modified Bessel function K0(x) for positive real x.
      REAL bessi0
      DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,
*         q2,q3,q4,q5,q6,q7,y                 Accumulate polynomials in double precision.
      SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
      DATA p1,p2,p3,p4,p5,p6,p7/-0.57721566d0,0.42278420d0,0.23069756d0,
*         0.3488590d-1,0.262698d-2,0.10750d-3,0.74d-5/
      DATA q1,q2,q3,q4,q5,q6,q7/1.25331414d0,-0.7832358d-1,0.2189568d-1,
*         -0.1062446d-1,0.587872d-2,-0.251540d-2,0.53208d-3/
      if (x.le.2.0) then                      Polynomial fit.
          y=x*x/4.0
          bessk0=(-log(x/2.0)*bessi0(x))+(p1+y*(p2+y*(p3+
*                 y*(p4+y*(p5+y*(p6+y*p7))))))
      else
          y=(2.0/x)
          bessk0=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+
*                 y*(q4+y*(q5+y*(q6+y*q7))))))
      endif
      return
      END
```

```
      FUNCTION bessi1(x)
      REAL bessi1,x
          Returns the modified Bessel function I1(x) for any real x.
      REAL ax
      DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,
*         q8,q9,y                       Accumulate polynomials in double precision.
      SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
      DATA p1,p2,p3,p4,p5,p6,p7/0.5d0,0.87890594d0,0.51498869d0,
*         0.15084934d0,0.2658733d-1,0.301532d-2,0.32411d-3/
      DATA q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,-0.3988024d-1,
*         -0.362018d-2,0.163801d-2,-0.1031555d-1,0.2282967d-1,
*         -0.2895312d-1,0.1787654d-1,-0.420059d-2/
      if (abs(x).lt.3.75) then          Polynomial fit.
          y=(x/3.75)**2
          bessi1=x*(p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7))))))
      else
          ax=abs(x)
          y=3.75/ax
          bessi1=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4+
*             y*(q5+y*(q6+y*(q7+y*(q8+y*q9))))))))
          if(x.lt.0.)bessi1=-bessi1
      endif
      return
      END
```

```
      FUNCTION bessk1(x)
      REAL bessk1,x
C     USES bessi1
         Returns the modified Bessel function K1(x) for positive real x.
      REAL bessi1
      DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,
     *     q2,q3,q4,q5,q6,q7,y           Accumulate polynomials in double precision.
      SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
      DATA p1,p2,p3,p4,p5,p6,p7/1.0d0,0.15443144d0,-0.67278579d0,
     *     -0.18156897d0,-0.1919402d-1,-0.110404d-2,-0.4686d-4/
      DATA q1,q2,q3,q4,q5,q6,q7/1.25331414d0,0.23498619d0,-0.3655620d-1,
     *     0.1504268d-1,-0.780353d-2,0.325614d-2,-0.68245d-3/
      if (x.le.2.0) then                Polynomial fit.
          y=x*x/4.0
          bessk1=(log(x/2.0)*bessi1(x))+(1.0/x)*(p1+y*(p2+
     *         y*(p3+y*(p4+y*(p5+y*(p6+y*p7))))))
      else
          y=2.0/x
          bessk1=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+
     *         y*(q4+y*(q5+y*(q6+y*q7))))))
      endif
      return
      END
```

The recurrence relation for $I_n(x)$ and $K_n(x)$ is the same as that for $J_n(x)$ and $Y_n(x)$ provided that ix is substituted for x. This has the effect of changing a sign in the relation,

$$\begin{aligned} I_{n+1}(x) &= -\left(\frac{2n}{x}\right) I_n(x) + I_{n-1}(x) \\ K_{n+1}(x) &= +\left(\frac{2n}{x}\right) K_n(x) + K_{n-1}(x) \end{aligned} \tag{6.6.4}$$

These relations are always *unstable* for upward recurrence. For K_n, itself growing, this presents no problem. For I_n, however, the strategy of downward recursion is therefore required once again, and the starting point for the recursion may be chosen in the same manner as for the routine bessj. The only fundamental difference is that the normalization formula for $I_n(x)$ has an alternating minus sign in successive terms, which again arises from the substitution of ix for x in the formula used previously for J_n

$$1 = I_0(x) - 2I_2(x) + 2I_4(x) - 2I_6(x) + \cdots \tag{6.6.5}$$

In fact, we prefer simply to normalize with a call to bessi0.

With this simple modification, the recursion routines bessj and bessy become the new routines bessi and bessk:

```
      FUNCTION bessk(n,x)
      INTEGER n
      REAL bessk,x
C     USES bessk0,bessk1
         Returns the modified Bessel function Kn(x) for positive x and n ≥ 2.
      INTEGER j
      REAL bk,bkm,bkp,tox,bessk0,bessk1
      if (n.lt.2) pause 'bad argument n in bessk'
      tox=2.0/x
```

```
      bkm=bessk0(x)                     Upward recurrence for all x...
      bk=bessk1(x)
      do 11 j=1,n-1                     ...and here it is.
          bkp=bkm+j*tox*bk
          bkm=bk
          bk=bkp
      enddo 11
      bessk=bk
      return
      END
```

```
      FUNCTION bessi(n,x)
      INTEGER n,IACC
      REAL bessi,x,BIGNO,BIGNI
      PARAMETER (IACC=40,BIGNO=1.0e10,BIGNI=1.0e-10)
C     USES bessi0
         Returns the modified Bessel function I_n(x) for any real x and n >= 2.
      INTEGER j,m
      REAL bi,bim,bip,tox,bessi0
      if (n.lt.2) pause 'bad argument n in bessi'
      if (x.eq.0.) then
          bessi=0.
      else
          tox=2.0/abs(x)
          bip=0.0
          bi=1.0
          bessi=0.
          m=2*((n+int(sqrt(float(IACC*n)))))     Downward recurrence from even m.
          do 11 j=m,1,-1                         Make IACC larger to increase accuracy.
              bim=bip+float(j)*tox*bi            The downward recurrence.
              bip=bi
              bi=bim
              if (abs(bi).gt.BIGNO) then         Renormalize to prevent overflows.
                  bessi=bessi*BIGNI
                  bi=bi*BIGNI
                  bip=bip*BIGNI
              endif
              if (j.eq.n) bessi=bip
          enddo 11
          bessi=bessi*bessi0(x)/bi               Normalize with bessi0.
          if (x.lt.0..and.mod(n,2).eq.1) bessi=-bessi
      endif
      return
      END
```

CITED REFERENCES AND FURTHER READING:

Abramowitz, M., and Stegun, I.A. 1964, *Handbook of Mathematical Functions*, Applied Mathematics Series, Volume 55 (Washington: National Bureau of Standards; reprinted 1968 by Dover Publications, New York), §9.8. [1]

Carrier, G.F., Krook, M. and Pearson, C.E. 1966, *Functions of a Complex Variable* (New York: McGraw-Hill), pp. 220ff.